```
# -*- coding: utf-8 -*-
"""
Example of how to use ObsPy to tune STA/LTA parameters for the Red
swarm on 2009/03/22, and extract event start and end times, and co
traces. A short extra step would allow these traces to be imported
GISMO as waveform and Catalog objects.

Created on Sat Apr 23 10:47:57 2016

@author: glennthompson
"""

# Load 1 day of data, trim to 5 minutes, plot time series & spectr
import sys
sys.path.append('/Users/glennthompson/Dropbox/scratch_matlab')
import tune_sta_lta as tsl
from obspy.core import read
import obspy.signal.trigger as trigger
from obspy.core.utcdatetime import UTCDateTime
tstart = UTCDateTime(2009, 3, 22, 3, 55, 0)
tend = UTCDateTime(2009, 3, 22, 4, 0, 0)
st = read("/Users/glennthompson/Dropbox/scratch_matlab/SEEDDATA/R*
st.plot(type='relative', equal_scale=False)
st.spectrogram()

# Filter from 0.8-12 Hz & downsample by factor 2
st.filter('bandpass', freqmin=0.8, freqmax=12.0, corners=2, zeroph
st.decimate(factor=2, strict_length=False)
st.plot(type='relative', equal_scale=False)

# This is the main signal we are trying to capture - and the time
# will maximise the STA/LTA ratio in
t_signal_start = 90.0
t_signal_end = 130.0
st.plot(type='relative', equal_scale=False, starttime=tstart+t_sig

# Run the tuning function for each trace
algorithm = 'classic_sta_lta'
numtries = 30
sta_best = list()
lta_best = list()
for tr in st:
    result = tsl.tune_sta_lta(tr, algorithm, t_signal_start, t_sig
    sta_best.append(result[0])
    lta_best.append(result[1])
```

```
# Summarize best settings for each trace, and the geometrical mean
import scipy.stats.mstats as mstats
sta_gmean=mstats.gmean(sta_best)
lta_gmean=mstats.gmean(lta_best)
print sta_best
print lta_best
print "Best STA window = %.1f seconds, Best LTA window = %.1f seco

# Plot the STA/LTA ratio
thresh_on = 5
thresh_off = 2.5
df = st[0].stats.sampling_rate
for tr in st:
    staltaratio = trigger.classic_sta_lta(tr.data, int(sta_gmean *
    trigger.plot_trigger(tr, staltaratio, thresh_on, thresh_off)

# Create trigger events by applying these same best STA/LTA settin
triggers_per_event = 3
import re # for some dumb reason, coincidence trigger needs algori
algorithm_without_underlines = re.sub('_', '', algorithm)
trig = trigger.coincidence_trigger(algorithm_without_underlines, t
from pprint import pprint
pprint(trig)
print "Number of events detected = %d" % len(trig)

# Plot each trigger
pretrig = 5;
posttrig = 5;
count = 0
for thistrig in trig:
    count += 1
    print "Event %d at %s" % (count,thistrig['time'].isoformat())
    st2 = st.copy()
    st2.trim(starttime = thistrig['time'] - pretrig, endtime = thi
    st2.plot(type='relative', equal_scale=False)
```